REMARKS

These remarks are responsive to the Office Action, dated October 18, 2007. Currently claims 1, 3-17, 19-26 are pending with claims 1, 7, and 17 being independent. Claims 2 and 18 have been previously cancelled. Claim 10 is amended to accommodate Examiner's objections.

Objections

In the October 18, 2007 Office Action, the Examiner objected to claim 10 based on various informalities. Applicants amended claim 10 to accommodate Examiner's objections. Thus, these objections are now moot.

35 U.S.C. 103

In the October 18, 2007 Office Action, the Examiner rejected claims 1, 3-6, 17, and 19-25 under 35 U.S.C. 103(a) as being unpatentable in view of various combinations of U.S. Patent No. 5,778,395 to Whiting et al. (hereinafter, "Whiting"), U.S. Patent No. 6,560,615 to Zayas et al. (hereinafter, "Zayas"), U.S. Patent Publication No. 2003/0070001 to Belknap et al. (hereinafter, "Belknap"), and "Efficient Distributed Backup with Delta Compression" to Burns et al. (hereinafter, "Burns").

In the Office Action, the Examiner stated that Whiting discloses all elements of the claims except that it "does not teach and further configured to capture a snapshot of a set of the shares of data at a particular point in time and to maintain a list of modified and/or created files since a last snapshot occurred." (Office Action, page 3). The Examiner stated that Zayas discloses this element. The Examiner further stated that "it would have been obvious to one of ordinary skill in the art at the time the invention was made to have modified Whiting et al. by the teaching of Zayas et al., since Zayas et al. teaches 'insertion and removal of entries in the MFL are performed by the storage system. When the first of a file's data and metadata bits are turned

on, the storage system adds the file to the MFL. In this way a file is added only once to the MFL' (see 7:40-45)". (Office Action, page 4). Applicants respectfully disagree and traverse these rejections.

Claim 1 recites, *inter alia*, a data protection system including a fileserver configured to contain shares of data and to be connected with a repository, wherein the repository is configured to store a replica of a file, the fileserver includes: a filter driver operative to intercept input or output activity initiated by client file requests and further configured to capture a snapshot of a set of the shares of data at a particular point in time and to maintain a list of modified and/or created files since a last snapshot occurred, a file system in communication with the filter driver and operative to store client files, the filter driver is configured to capture the snapshot at a specified time interval based on a backup frequency defined in a protection policy stored in the fileserver.

As understood by Applicants, Whiting relates to a system for backing up files from disk volumes on multiple nodes of a computer network to a common random-access back storage means. (Whiting, Abstract). Whiting further relates to a lower cost backup solution. (Whiting, Col. 4, lines 63-64). Whiting's files are backed up from disk volumes on multiple nodes of a computer network to a common random-access backup storage means, typically a disk volume. (Whiting, Col. 5, lines 3-6). Whiting's backups can occur automatically or independently for each node and can be scheduled by the user or administrator. (Whiting, Col. 5, lines 6-8). This is different than the filter driver that is configured to capture the snapshot at a specified time interval based on a backup frequency defined in a protection policy stored in the fileserver, as recited in claim 1. Instead, Whiting performs backup operations of its based on indications from the user or administrator. Whiting does not include a filter driver that captures the snapshot at a

specific period of time based on a backup frequency contrary to the teachings of the present invention. As such, contrary to the Examiner's suggestions, Whiting fails to disclose the filter driver element recited in claim 1.

Whiting's system creates four types of files during backup: new, unchanged, updated, and modified. (Whiting, Col. 7, lines 59-65). Each time a backup set of files is migrated, Whiting's solution searches for matching files for each new or updated file. (Whiting, Col. 8, lines 8-16). Further, during backup, Whiting creates and modifies files over network on the backup storage. (Whiting, Col. 7, lines 8-13). Each node in Whiting is assigned to two directories, a user directory and a system directory, on the backup storage means. (Whiting, Col. 7, lines 13-17). However, Whiting fails to disclose, teach or suggest a fileserver configured to contain shares of data and to be connected with a repository, wherein the repository is configured to store a replica of a file, as recited in amended claim 1. In contrast, Whiting only creates and modifies files on the backup storage. This is different from present invention's a fileserver connected to a repository that stores a replica of a file. Whiting only deals with four categories of files: new, unchanged, updated or modified, but fails to teach file replicas being stored on repositories, as in the present invention.

Whiting's client systems (e.g., desktops, laptops, server, etc.) are the source of backup data and a fileserver acts as a destination of backup data. Whiting maintains a replica of data from client systems that are spread out over the network into their destination fileserver. This is in contrast to the present invention, where the fileserver is the source of backup and a repository is the destination of backup. The present invention maintains a replica of all fileserver data in a repository. Thus, the fileserver and the repository are parts of the system in the present invention.

Further, Whiting does not include a filter driver that is logging changes to a file as they happen. Whiting fails to disclose how files are subdivided into the four categories mentioned above. (Whiting, Col. 7, line 61). This in contrast to the present invention that continually tracks all changes to the file system located on the fileserver as the files are modified so that when a snapshot is taken, there is a complete list of the files that have to be backed up available. Thus, there is no need to look at every file in the file system, i.e., "walk the file system". Whiting fails to disclose how the determination is made as to which files must be backed up. It appears that Whiting's system is similar to the conventional backup applications that simply "walks the file system" for any changes. Thus, Whiting fails to disclose, teach or suggest this element of claim 1.

Zayas fails to cure the deficiencies of Whiting. As understood by Applicants, Zayas relates backup data and techniques for speeding up backup operations. (Zayas, Col. 1, lines 9-10). When Zayas creates a volume of files, a Modified Files List ("MFL") is established and an epoch timestamp (identifying an important point in time for the volume) is set. (Zayas, Col. 3, lines 38-40). Each leafnode in Zayas' MFL stores a file identifier (file ID) and an epoch timestamp. The file ID identifies a file on the volume that has been modified since it was last archived. (Zayas, Col. 5, lines 31-34). Zayas' epoch timestamp identifies the first epoch in which the file identified by file ID was modified since it was last archived. (Zayas, Col. 5, lines 38-40). Zayas further enumerates and orders all **identified** files in the MFL that were first modified before the selected epoch. (Zayas, Col. 7, lines 16-18). (emphasis supplied). The enumerated and ordered files are archived. (Zayas, Col. 7, lines 25-26). Once the files are archived, their data and metadata bits are turned off; the entries for these files are removed from the MFL. (Zayas, Col. 7, lines 34-36). This is different than having a filter driver operative to intercept input or output

activity initiated by client file requests and further configured to capture a snapshot of a set of the shares of data at a particular point in time and maintain a list of modified and/or created files since last snapshot occurred, as recited in claim 1. Instead, Zayas provides its files with a file ID and a timestamp that identify the file and whether the file was modified since it was last archived. The present invention captures a snapshot of an entire file system at a specific point in time. Such snapshots are taken at specific periods of time and lists of files are maintained since last snapshot and not since last archive, as taught by Zayas. Further, Zayas appears to enumerate and order only specific **identified** files, rather than modified and/or created files of claim 1. (emphasis supplied). These identified files only relate to those files that were modified prior to the selected epoch (i.e., period of time). Once Zayas archives the files, entries corresponding to the files are removed from the modified file list. In contrast, the present invention captures a snapshot of a set of shares of data at a particular point in time, rather than capturing specific identified files. Further, the present invention maintains a list of files since last snapshot and does not remove any files from the lists.

Further, Zayas' MFL is a persistent metadata storage structure that is maintained for the life of a volume contained on the storage structure. The present invention's list of new and modified files is maintained for just a single backup period and then the modified file list and the snapshot are deleted. Zayas fails to teach a filter driver implementation for updating the MFL. Zayas' MFL is a persistent metadata storage structure that is maintained for the life of a volume. The present invention's list of new and modified files is maintained from one snapshot to the next and then deleted.

Additionally, Zayas discusses end user primary storage data being deleted and the subsequent ramifications of that deletion on Zayas' MFL, which is irrelevant to backup retention

management. In contrast, in the present invention, an end-user deletion activities (similar to those discussed in Zayas) have no effect on the retention of data in the repository. The present invention's retention policy defines just how much backup history will be maintained. For example, many conventional IT organizations maintain 13 weeks of backup data to be able to recover data to any point within that 13 week period. As another example of conventional backup retention, a company may be under a federal regulation to retain all data for 7 years. Zayas fails to address these issues with backup retention, whereas the present invention implements such retention policies. As such, Zayas fails to disclose the filter driver element of claim 1.

The improper combination of Whiting and Zayas fails to realize the present invention. The combination of Whiting and Zayas results in a system that allows stamping of identified files in the four categories of files, e.g., new, unchanged, updated, and modified, with a file ID and an epoch time stamp, indicating time since prior backup, so that the files can be enumerated and ordered and then their entries in a Modified File List can be removed. Clearly, this is different than what is recited by the present invention's claim 1. Specifically, the improper combination of Whiting and Zayas fails to disclose, teach or suggest, *inter alia*, a fileserver configured to contain shares of data and to be connected with a repository, wherein the repository is configured to store a replica of a file, the fileserver includes: a filter driver operative to intercept input or output activity initiated by client file requests and further configured to capture a snapshot of a set of the shares of data at a particular point in time and to maintain a list of modified and/or created files since a last snapshot occurred, the filter driver is configured to capture the snapshot at a specified time interval based on a backup frequency defined in a protection policy stored in the fileserver.

One of the advantages of the present invention is that it represents two components, one or more fileservers acting as a primary storage NAS tier and one or more repository servers acting as a back end backup repository for these fileservers. There are no backup clients and agents out on the network that are part of this invention. This is similar to an integrated primary storage system that backs itself up into a grid-based backup repository. Additionally, in the present invention, the most active files of a fileserver are maintained in their entirety on the fileserver. When a fileserver's file system gets close to running out of available space, certain files that have not been accessed for the longest time are stubbed out on the fileserver. One of the major differences between Whiting and the present invention is that the present invention leverages backup data it already has to act as a second tier of data storage. Effectively the same data in the repository that represents backup history over time is also leveraged to represent a second tier of storage for inactive files of a fileserver. The fileserver can operate with either a high caching level or low-caching level, based on the protection policy for each share on that fileserver. As a fileserver providing data to networked NAS clients, a high caching level suits a read-mostly file model and assures the fastest response time to users. A low caching level is used for write-mostly applications like a backup application that is writing to a fileserver in order for the fileserver to backup that data into the backend repository.

Another advantage of the present invention is that it integrates backup and HSM, leveraging the most recent version of the data that's already been backed up as the second tier of storage to the fileserver's primary tier. In contrast, Whiting treats these two data management applications as separate entities with backup data going to a disk repository and archive/HSM data going to tape or optical disk.

Thus, neither Whiting, nor Zayas, nor their combination disclose, teach, or suggest all elements of claim 1, and claim 1 should be allowed.

Further, Belknap does not cure the deficiencies of the combination of Whiting and Zayas. Belknap discloses a media manager which incorporates an application program interface (API) for converting high-level generic commands into device-level commands for output to a media device. (Belknap, Abstract). Further, Belknap determines whether a media object is located within a multimedia data storage system by searching an index of media objects stored within the system. (Belknap, para. [0063]-[0064]). Belknap fails to disclose, teach or suggest, *inter alia*, a filter driver operative to intercept input or output activity initiated by client file requests and further configured to capture a snapshot of a set of the shares of data at a particular point in time and to maintain a list of modified and/or created files since a last snapshot occurred; the filter driver is configured to capture the snapshot at a specified time interval based on a backup frequency defined in a protection policy stored in the fileserver, as recited in claim 1.
In Therrien's invention, the final repository node location for the backup of each file is determined automatically and virtually across multiple repository nodes. This is performed by a sophisticated location manager and location cache to eliminate the need of an administrator or end-user to define the exact and specific location of a backup file. This location cache and location manager allows multiple repository nodes to be capacity balanced and act as a single larger pool of disk storage.

Further, Belknap discloses an audio/video file media manager. For data directed to removable storage media, the media manager tracks which audio/video file was recorded on each medium (tape, optical disk, etc.). This is similar to a conventional backup catalog function. In contrast, the present invention stores backup data on one or more repository servers that have

non-removable hard disk storage. The purpose of the location manager is not only to locate a file from among multiple repository nodes. In addition, it allows backup data in one repository server to be moved to another repository server for the purpose of load balancing for performance or capacity. The location cache is an in-memory image of the current location of all objects to be stored. It provides for accelerated lookups to the location of backup files among repository servers. Belknap fails to disclose a location cache.

Instead, Belknap's invention provides an abstraction layer to allow any kind of object storage device to be used for storing media files (audio/video). This is different than the present invention that deals with homogeneous server/magnetic disk drive systems that can be load-balanced over time. The location cache provides a level of indirection that allows data stored within a specific repository node to be moved to a different node under circumstances of cross-repository-server capacity or performance imbalance.

Also, Burns fails to cure the deficiencies of Whiting and Zayas combination as well. Burns relates to constraining client-side differencing and two tape accesses for delta restore and eliminates the use of reverse delta chains. (Burns, Section 4.2). In order to update a single version in the reverse delta chain, Burns' server must store two new files, recall one old file, and perform both the differencing and reconstruction operations. (Burns, Section 4.2). However, Burns fails to disclose, teach or suggest, *inter alia*, a filter driver operative to intercept input or output activity initiated by client file requests and further configured to capture a snapshot of a set of the shares of data at a particular point in time and to maintain a list of modified and/or created files since a last snapshot occurred, the filter driver is configured to capture the snapshot at a specified time interval based on a backup frequency defined in a protection policy stored in the fileserver, as recited in claim 1.

Thus, the various combinations of Whiting, Zayas, Belknap, and Burns do not render claim 1 obvious. As such, the rejection of claim 1 is respectfully traversed. Applicants respectfully request that the rejections of claim 1 are withdrawn and claim 1 is allowed.

Claims 3-6, 17, and 19-25 are not rendered obvious by the various combinations of Whiting, Zayas, Belknap, and Burns for at least the reasons stated above with regard to claim 1. As such, the rejections of claims 3-6, 17, and 19-25 are respectfully traversed. The Examiner is requested to reconsider and withdraw his rejection of claims 3-6, 17, and 19-25.

In the October 18, 2007 Office Action, the Examiner rejected claims 7-16, and 26 under 35 U.S.C. 103(a) as being unpatentable over various combinations of U.S. Patent No. 6,847,982 to Parker et al. (hereinafter, "Parker"), Zayas, "Deciding when to forget in the Elephant file system" to Santry et al. (hereinafter, "Santry").

In the Office Action, the Examiner stated that Parker discloses all elements of claim 7 except that it fails to teach "capturing a snapshot of the set of files at a particular point in time." (Office Action, page 10). It also appears that Examiner alleges that "[m]aintaining a list of modified and/or created files since last captured snapshot" is also taught by Zayas and not by Parker. (Office Action, page 10). The Examiner further stated that "it would have been obvious to one of ordinary skill in the art at the time the invention was made to have modified Parker et al. by the teaching of Zayas et al., since Zayas et al. teaches 'insertion and removal of entries in the MFL are performed by the storage system. When the first of a file's data and metadata bits are turned on, the storage system adds the file to the MFL. In this way a file is added only once to the MFL' (see 7:40-45)". (Office Action, page 10). Applicants respectfully disagree and traverse these rejections.

Claim 7 recites, *inter alia*, a method for protecting data including storing a version of a file within a set of files on a primary disk storage system, capturing a snapshot of the set of files at a particular point in time based on a backup frequency defined in a protection policy, maintaining a list of modified and/or created files since last captured snapshot, examining the protection policy associated with the set of files to determine where and how to protect files associated with the set of files, and replicating the version of the file to a repository specified by the protection policy, wherein the repository includes at least one of a local repository and a remote repository.

As understood by Applicants, Parker discloses an intelligent data inventory and asset management software system. (Parker, Col. 7, lines 18-23). The Parker system includes an Akashic File Clerk that maintains an inventory database, which includes electronic signatures for every file on a work station and all new and changed files. (Parker, Col. 7, line 24-28). Parker allows a client to determine which files are critical and which are not critical, then Parker runs inventories to capture the files that have changed and forwards the changed files to an Akashic Vault for storage and processing. (Parker, Col. 7, lines 28-35). During inventories, Parker identifies files that have 1) changed since the last inventory, 2) been deleted since the last inventory, 3) been added since the last inventory. (Parker, Col. 8, lines 17-26). As agreed by the Examiner, Parker does not capture a snapshot of the set of files at a particular point in time based on a backup frequency defined in a protection policy and maintain a list of modified and/or created files since last captured snapshot, as recited in claim 7. Instead, Parker's Akashic File Clerk that contains signatures of files, runs an inventory on the changed files, and then stores the files in the Akashic Vault, but does capture a snapshot of the set of files nor does it examine a protection policy and determine where and how to protect files, as recited in amended claim 7.

Parker's Akashic Vault is a computer that is attached as a node to the client's network which stores captured files. (Parker, Col. 7, lines 44-46). After capturing files, Parker's Vault generates reverse and forward deltas, then deletes the previous version and archives the newest compressed version of the file. (Parker, Col. 9, line 54 to Col. 10, line 4). Parker generates a list of forward delta(s) and copies of the new files and sends them to an offsite Library System. (Parker, Col. 10, lines 5-8). This is different than replicating the version of the file to a repository specified by the protection policy, wherein the repository includes at least one of a local repository and a remote repository, as recited in claim 7.

Parker discloses how files at a client system are check-summed to determine whether their content changes over time and only those files that are new or have changed are sent to the Akashic Vault. (Parker, Col. 7, lines 24-35). Parker doesn't deal with storing historical versions of data over time. This is different than the present invention where a filter driver monitors a file system to determine which files have been created, modified or deleted since the last backup. Further, Parker's replication source and destination are not peers in terms of performance/cost /reliability. Parker's Akashic Vault is a local storage system that appears to be based on magnetic disk technology. Parker's offsite Library System appears to be an optical "jukebox" with a unique write-once properties. Parker demonstrates another example of a conventional tiered storage system where the time it takes to recover data from the Library system is much greater than the time it takes to recover from the local vault. This is different than the present invention, one of the advantages of which is that it supports a local and remote repository with identical storage capacity/performance/cost and reliability – servers with storage.

Additionally, Parker must maintain a database of the running history of all files and how this database helps to determine which files have been changed, deleted or added. (Parker, Col. 8, lines 17-28). In present invention, by tracking all file system create, write and delete operations in real time, there is no need to maintain a database of what's been backed, in contrast to Parker. From backup to backup, the present invention monitors these changes to know what to backup at the next backup time. As such, Parker does not disclose all elements of claim 7 and claim 7 should be allowed.

Zayas fails to cure the deficiencies of Parker for at least the reasons stated above with regard to claim 1. Specifically, Zayas fails to disclose, teach or suggest, *inter alia*, capturing a snapshot of the set of files at a particular point in time based on a backup frequency defined in a protection policy and maintaining a list of modified and/or created files since last captured snapshot, as recited in claim 7.

The improper combination of Parker and Zayas fails to realize the present invention. The combination of Parker and Zayas results in a system that allows stamping of specific files that were identified in an inventory with a file ID and an epoch time stamp, indicating time since prior backup, so that the files can be enumerated and ordered and then their entries in a Modified File List can be removed. Clearly, this is different than what is recited by the present invention's claim 1. Specifically, the improper combination of Parker and Zayas fails to disclose, teach or suggest, *inter alia*, capturing a snapshot of the set of files at a particular point in time based on a backup frequency defined in a protection policy and maintaining a list of modified and/or created files since last captured snapshot, as recited in claim 7.

Santry also fails to cure the deficiencies of either Parker, Zayas, or their combination. As understood by Applicants, Santry discloses a file system that keeps old versions of the file for

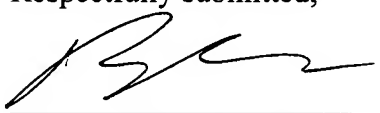
recovery purposes (Santry, pg. 111, section 1). In some instances, Santry does not keep old versions of the files and only keeps a single current version. (Santry, pg. 113, section 3.3). However, neither Parker, nor Zayas, nor Santry nor their combination discloses, teaches or suggests the subject matter of claim 7. As such, the rejection of claim 7 is respectfully traversed and the Examiner is requested to reconsider and withdraw his rejection of claim 7

Claims 8-16 and 26 are patentable over various combinations of Parker, Zayas, and Santry for at least the reasons stated above with regard to claim 7. As such, the rejections of claims 8-16 and 26 are respectfully traversed. The Examiner is requested to reconsider and withdraw his rejections of claim 8-16 and 26.

No new matter has been added. The claims currently presented are proper and definite. Allowance is accordingly in order and respectfully requested. However, should the Examiner deem that further clarification of the record is in order, we invite a telephone call to the Applicants' undersigned attorney to expedite further processing of the application to allowance.

Applicants believe that no additional fees are due with the filing of this Amendment. However, if any additional fees are required or if any funds are due, the USPTO is authorized to charge or credit Deposit Account Number: 50-0311, Customer Number: 35437, Reference Number: 25452-013.

Respectfully submitted,



Date: February 6, 2008

Boris A. Matvenko, Reg. No. 48,165
MINTZ LEVIN COHN FERRIS
GLOVSKY & POPEO, P.C.
Chrysler Center
666 Third Avenue, 24th Floor
New York, NY 10017
Tel: (212) 935-3000
Fax: (212) 983-3115